PHOS Board Controller Specification

Document name: *PHOS BC specification_v3.4.doc*

Revision: *0.6 (fw version 3.4)*

Date: *Created on 5/23/2007 9:38:00 AM*

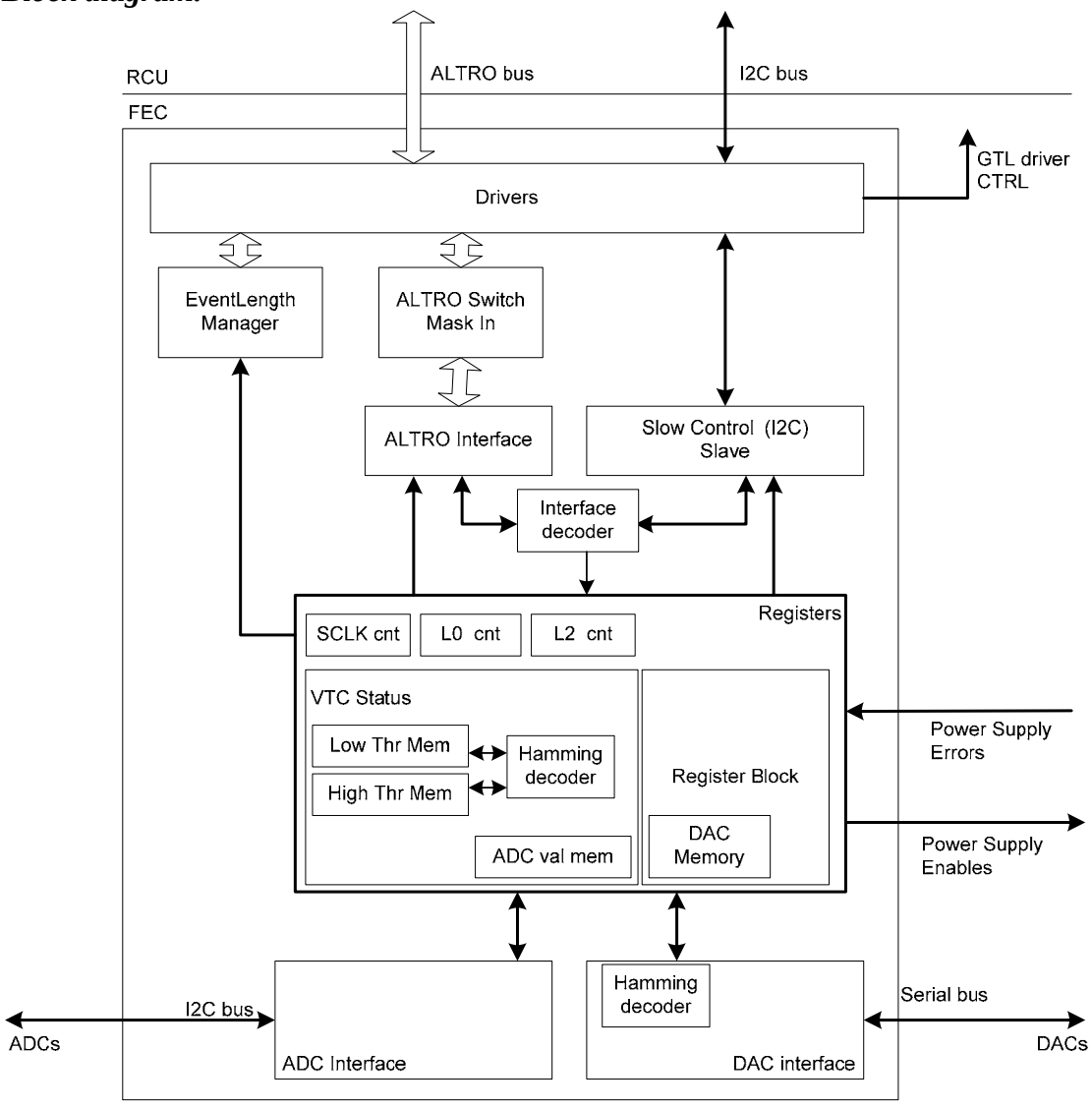
Last saved: 10/31/2007 11:12:00 AM

Author: *Created by Johan Alme*

Last saved by Johan Alme

Module: ***PHOS BC***

Block diagram:



Features:

- Two command interfaces
  - ALTRO bus interface
  - Special I2C interface
- I2C interface is robust with timeout counters and masking of bus input when active transaction to other card.
- Setting of DACs for bias voltage for High Voltage region
- Interface to 3 ADCs for verifying voltage and current-levels as well as temperatures.
- Programmable min/and max thresholds for flagging errors in ADC values.
- Possible to set how to treat the voltages read back for current calculations:
  - Store them as is
  - diff with previous adc value.
  - Control the order of diff after the expected direction of the current
- Unlock register for write to read-only registers
- Monitoring error inputs from Power Regulators
- Interrupt line to RCU for errors of a severity level craving urgent measures
- Configurable number of threshold violations from 1 to 3 before interrupt is flagged.
- Possible to set the FEC in standby mode by turning off voltage regulators and not reporting any warnings/errors.
- Radiation precautions:
  - Hamming coded ADC threshold settings
  - Hamming coded DAC values
  - TMR of configuration/status registers and threshold config registers
- Configurable automatic update of DACs
- Event-length Manager module that generates a hitmap of all ALTRO channels and clocks this out to RCU by controlling dstb for Sparse Readout Mode.
- Debug registers to test ALTRO bus integrity

1 DOCUMENT CONTROL 4
1.1 REVISION HISTORY 4
1.2 REFERENCES 4
2 MOTIVATION 5
3 EXTERNAL INTERFACE 6
3.1 SIGNAL INTERFACE 6
3.2 TIMING DIAGRAMS 9
4 REGISTER INTERFACE 10
5 FUNCTIONAL REQUIREMENTS 17
5.1 FUNCTIONAL OVERVIEW 17
5.2 MAIN FUNCTIONAL CHANGES FROM PCM V2.0 (HUST) 17
5.3 PROJECT SETUP 17
5.3.1 Software 19
5.4 FUNCTIONAL DETAILS 20
5.4.1 Main Output Signals 20
5.4.2 Drivers (Glue Logic) 20
5.4.3 ALTRO Switch Mask In 20
5.4.4 ALTRO Interface 20
5.4.5 Event-length Manager 21
5.4.6 Slow Control Slave Interface 22
5.4.7 Interface Decoder 23
5.4.8 Registers 23
5.4.9 ADC Interface 24
5.4.10 DAC Interface 25
5.4.11 Hamming Code / Hamming Decoder Module 26
5.4.12 Optional Functionality 27
6 OTHER REQUIREMENTS 28
6.1 CLOCK STRATEGY 28
6.2 RESET STRATEGY 28
6.3 POWER STRATEGY 28
6.4 TEST STRATEGY 28
6.4.1 Functional and Post Place and Route Verification 28
6.4.2 Functional Coverage 29
6.4.3 Hardware Verification 29
7 PHYSICAL IMPLEMENTATION 30
7.1 TECHNOLOGY 30
7.2 LOGIC SYNTHESIS 30
7.2.1 Static timing analysis 30
7.2.2 Area estimates 30
8 INSTALLATION LOG 31
8.1 INSTALLATION STATUS 31
8.2 SOFTWARE 31

1 Document control

1.1 Revision history

Rev.	Rev. date	Document status	Responsible
0.1	24.05.07	First draft	JA
0.2	16.08.07	Updated after design of v3.0	JA
0.3	04.09.07	Updated to match version 3.1 of firmware	JA
0.4	10.10.07	Updated to match version 3.2 of firmware	JA
0.5	16.10.07	Updated to match version 3.3 of firmware	JA
0.5	30.10.07	Updated to match version 3.4 of firmware	JA

1.2 References

Ref. No.	Doc. Name.	Rev / Rev date	Title
1.	PCM_2_0.pdf	Ver 2.0, Aug 2006	PCM 2.0 8Based on PHOSS FEE board controller 0.1 (HUST)
2.	AD7416_AD7417_7418.pdf	Rev. G [2004]	Analog Devices 10-Bit Digital Temperature Sensor (AD7416) and Four Single-Channel ADCs (AD7417/AD7418)
3.	FEE-V1.1b.pdf	V1.1 27.Sep 05	PHOS FEE v1.1 Schematics
4.	PHOS-User-Manual.pdf	Rev 2.1 4. Jan 07	PHOS Basics for the User
5.	MAX5308-MAX5309.pdf	Rev 0; 8/01	Maxim Low-Power, Low-Glitch, Octal 10-Bit Voltage-Output DACs with Serial Interface
6.	TPC-BC_v2.3.pdf	9/12/2004	ALICE TPC Board Controller (version 2.3)
7.	acex.pdf	Ver 3.4, May 03	ALTERA Acex 1K Programmable Logic Device Family

2 Motivation

The Front End Electronics in PHOS is consisting of one Readout Control Unit (RCU) and 28 Front End Cards (FECs) connected to the RCU via two separate branches. On each FEC an SRAM based FPGA is situated – the Board Controller. Since the Front End Electronics is physically unavailable when PHOS is fully commissioned it must be possible to check the status via software during operation, and quickly respond to any error situation that might occur.

The purpose of the Board Controller is to read crucial values on the FEC, such as voltages, currents and temperatures. If these values exceed given programmable thresholds the RCU will be notified. If the severity level is considered to be potentially damaging to the board, the RCU will turn off the given FEC.

The PHOS FEC also includes a high voltage section. A very important functionality of the PHOS Board Controller is to set the bias voltage to the charge sensitive amplifiers located in the high voltage section. This must be done since the amplification of the APDs that are used for readout is varying from part to part. This variation is cancelled out with the setting of the DACs that are controlling the bias voltage.

This version of the PHOS BC is based on the TPC BC, adding the extra functionality needed for PHOS, and removing some features that are not needed. The basis for the code is the FMD version that is a VHDL implementation of the TPC BC with some modifications. Major changes have been done to this implementation to make it fit for PHOS and to make it more robust.

3 External Interface

3.1 Signal interface

Signal name	Dir	Pin	Sync[1]	Description
rdoclk	in	183		40 Mhz Readout Clock (used as system clock)
rdoclk_en	out	121		Enable for the ALTRO readout clock
fecclk_40m	in	79		40 MHz Clock from crystal
adcclk	in	167		20 MHz ALTRO sampling clock.
adcclk_en	out	122		Enable signal for the ALTRO sampling clock
rst_fbc	out	101		Reset from RCU (command decoded) – active low
sys_rst	in	80		Global reset signal from power up
bd[39:0]	Inout	[40,41, 44,45, 46,47, 53,54, 55,56, 57,58, 60,61, 63,64, 65,67, 68,69, 70,71, 73,74, 75,83, 85,86, 87,88, 89,90, 92,93, 94,95, 96,97, 99,100]		Bidirectional ALTRO bus: [39] Parity bit [38] Bcast bit [37] Boardcontroller/ALTRO [36:25] Channel Address / BC register address [24:20] ALTRO Instruction Code [19:0] Data
write	Inout	197		The write/read signal is driven by the master (RCU) and defines whether the access to the addressed unit is in write/read mode (low/high).
cstb	Inout	104		The master (RCU) drives the command strobe (CSTB) signal. When asserted, it indicates that a valid word has been placed in the AD bus. The signal also qualifies the WRITE signal.
ackn	inout	205		On a WRITE or COMMAND cycle, the addressed unit asserts the ACKN signal to indicate that is has successfully latched the bus content and executed the requested instruction. On a READ cycle, the addressed unit asserts the ACKN to indicate that it has placed the requested data on the bus.
error	inout	37		The ERROR line is asserted by the slave units to signal the occurrence of an error condition. If the error condition has occurred in an instruction cycle (parity error or

[1] Inputs: <clock name>|async: Assumed synchronous with the given clock name or asynchronous
Ouputs: <clock_name>[glitch free]|comb: Output generated from the given clock name or combinatorial

				instruction code error), the slave does not acknowledge the instruction cycle and asserts the ERROR signal.
trsf	inout	39		The ALTRO chip takes the control of the bus by asserting the TRANSFER signal acknowledges this instruction cycle. TRANSFER is kept asserted till the data block has been completely transferred.
dstb	inout	38		The data transfer is not necessarily continuous and for this reason each single word, being transferred, is validated by the signal DSTB (Data Strobe).
l1	In	102		The l1 signal is broadcasted by the RCU to all the FECs. It is used for the distribution of the trigger information. The l1 signal is synchronous with the SCLK signal and lasts for at least two clock cycles.
l2	in	103		The l2 signal is broadcasted by the RCU to all the FECs. It is used for the distribution of the trigger information. The l2 signal is synchronous with the RCLK and lasts for two clock cycles.
trsf_en	In	195		trsf_en is used to drive the bi-directional bus bd when transferring an event.
ack_en	In	193		ack_en frames ackn, enabling the intrinsic capacitor in the transceiver.
dolo_en	In	186		dolo_en is used to drive the bi-directional bus bd when reading a register for the later.
card_ad[4:0]	in	[176, 175, 174, 173, 172]		Hardware address input from branch
bcout_add[4:0]	out	[192, 191, 190, 189, 187]		Output hardware address to the ALTROs: “00000” if ALTROs are turned off or in debug mode card_ad otherwise.
altrops_en	out	126		Enable signal for ALTRO power regulators
biasps_en	out	128		Enable signal for bias power regulators
shaperps_en	out	127		Enable signal for shaper power regulators
allps_error	in	78		Error flag signalling if output voltage has dropped 5% under nominal value for the power regulator of the digital part.
altrops_error	in	180		Error flag signalling if output voltage has dropped 5% under the nominal value for ALTRO power regulator. Low parity
biasps_error	in	168		Error flag signalling if output voltage has dropped 5% under the nominal value for bias power regulator. Low parity
shaperps_error	in	163		Error flag signalling if output voltage has dropped 5% under the nominal value for shaper power regulator. Low parity
oeab_l	out	202		Setting direction of GTL drivers for altro bus: 0: output (to RCU) 1: input (to FEC)
oeab_h	out	160		Setting direction of GTL drivers for altro bus: 0: output (to RCU) 1: input (to FEC)

oeba_l	out	200		Setting direction of GTL drivers for altro bus: 1: output (to RCU) 0: input (to FEC)
oeba_h	out	196		Setting direction of GTL drivers for altro bus: 1: output (to RCU) 0: input (to FEC)
ctr_in	out	198		Enabling output of GTL drivers for control bus (RCU->FEC): 0: enabled 1: high Z
ctr_out	out	199		Enabling output of GTL drivers for control bus (FEC -> RCU): 0: enabled 1: high Z
rcu_scl	in	184		5 MHz Slow Clock for I2C transfers
rcu_sda_in	in	119		Slow Control Serial data in from RCU
rcu_sda_out	out	120		Slow Control Serial data out to RCU
bc_int	out	16		Interrupt line to RCU, Active Low.
sensor_scl	out	7		I2C clock for AD7417 communication (x3)
sensor_sda	inout	8		I2C data line for AD7417 communication (x3)
convst	out	13		Convert start signal for AD7417 (x3)
oti_1	In	9		Over-temperature indicator (Not used)
oti_2	In	11		Over-temperature indicator (Not used)
oti_3	In	12		Over-temperature indicator (Not Used)
dac_clk_en	out	10		DAQ serial clock enable. Active high
dac_sclk	out	17		DAC serial Clock
dac_ldac	out	203		Load DAC. LDAC is asynchronous active-low that updates the DAC outputs simultaneously. If LDAC is driven low, the DAC registers are transparent.
dac_sel[3:0]	out	[26, 25, 24, 18]		Chip-Select output (active-low) daq_sel[0] – HVB 1 – 8 daq_sel[1] – HVA 1 – 8 daq_sel[2] – HVA 9 – 16 daq_sel[3] – HVB 9 – 16
dac_din[3:0]	out	[30, 29, 28, 27]		Serial Data Input for the 4 dacs
dac_dout[3:0]	in	[36, 31, 15, 14]		Data Output from the 4 daqs. DOUT is updated on the falling edge of SCLK.
test_m_sclk_dn	out	166		LVDS_n Test mode sampling clock generated by the Board Controller. Can only be used of external SCLK is missing
test_m_sclk_dn	out	164		LVDS_p Test mode sampling clock generated by the Board Controller
test_m/g	in	157		Enable signal to enable test sampling clock and RCLK for BC. Active low.
tms_altro0	out	111		*Test mode Altro 0, active low (Not Used)*
tms_altro1	out	112		*Test mode Altro 4, active low (Not Used)*
tms_altro2	out	113		*Test mode Altro 2, active low (Not Used)*
tms_altro3	out	114		*Test mode Altro 3, active low (Not Used)*
adc_add[1:0]	out	[116, 115]		*Output addressing lines for testmode. Used to address which of the 4*4 channels in the ALTROS should be readout during testmode execution.* *"00" – ch 0-3, "01" – ch 4-7,* *"10" – ch 8-11, "11" – ch 12-15* *Not Used*
led_red	out	62		Output to diode (connected high)

led_yellow	out	179		Output to diode (connected high)
led_green	out	177		Output to diode (connected high)
aux[3:0]	out	[162, 161, 159, 158]		Output to diode (connected high)
usb_sda	inout	208		*USB interface - Not Used*
usb_scl	in	207		*USB interface - Not Used*
usb_ef	out	206		*USB interface - Not Used*
usb_rw	in	170		*USB interface - Not Used*
usb_cs	in	169		*USB interface - Not Used*
usb_fd[15:0]	inout	[150, 149, 148, 147, 144, 143, 142, 141, 140, 139, 136, 135, 134, 133, 132, 131]		*USB interface - Not Used*
usb_clr_fifo	out	125		*USB interface - Not Used*
usb_ifclk	in	182		*USB interface - Not Used*
usb_ff	out	204		*USB interface - Not Used*

Table 3-1: Signal interface

3.2 Timing diagrams

TBA

4 Register interface

Register name	Addr.	Type [2]	Allow Brcast	Description
UNLOCK	0x0	RW	Yes	One bit unlock register for writing to read only regs for testing. 0: Locked 1: Unlocked
L0CNT[15:0]	0x0B	R(W)	Yes*	Number of L0 triggers received
L2CNT[15:0]	0x0C	R(W)	Yes*	Number of L2a triggers received
SCLKCNT[15:0]	0x0D	R(W)	Yes*	Sampling clock counter
SLOWCTR_ERR [7:0]	0x0E	R	No	Number of timeout situations in Slow Control
CSR0[11:0]	0x11	RW	Yes	Interrupt Mask Register: *Default value = 0x1FF* [11] **HV Update Mode.** 0 = The BC updates DAC with update_hv command. 1 = The BC continuously updates DACs [10] **Conversion Mode.** 0 = The BC reads the content of the monitor ADC with the STCNV command (1B) 1 = monitor ADC converts continuously [9:8] **Error Mask**. These two bits mask the assertion of the Error line. This line is asserted with the flags registered in CSR1[9:8] 0 = the error is masked 1 = the error asserts the line [7:0] **Interrupt Mask.** These bits mask the bits of CSR1 [7:0] for the assertion of the Interrupt line
CSR1[13:0]	0x12	R	N/A	Error Status Register: *Default value = 0x0000* [13] Value of Slow Control Interrupt line [12] Value of ALTRO bus Error line [11] Slow Control Instruction Error [10] ALTRO error: Registered value of ALTRO bus error line [9] ALTRO bus Instruction Error (to BC) [8] Parity error of ALTRO bus 20 MSB [7] Missing Sampling Clock [6] ALTRO Power, Digital 4.2V & 3.3V and Bias Supply Error [5] Shaper +6.0V Power Supply Error [4] 4.2 V or 3.3 V digital current is higher than thresholds. [3] 4.2 V or 3.3 V digital voltage is lower than thresholds.

[2] Legend: W=write, R=read, T= write trigger (not physical registers)

Register name	Addr.	Type [2]	Allow Brcast	Description
				[2] 4.0 V, +6.0 V, -6.0 V, 13.5 V analog current higher than threshold. [1] 4.0 V, +6.0 V, -6.0 V, 13.5 V analog voltage lower than threshold. [0] Temp1, Temp2 or Temp3 higher than threshold.
CSR2[15:0]	0x13	RW	Yes	Status and Configuration *Default value = 0x013F* [15:11] Hardware Address (read only) [10] Card Isolated [9:8] Number of times a ADC threshold violation must occur before it is reported [7] *Not Used* [6] Enables Hamming correction on HVDAC values and Thresholds. [5] Enables DAC clock [4] Enables Sampling Clock [3] Enables Readout Clock [2] Power Switch for Shaper Power Regulator [1] Power switch for Bias Power Regulator [0] Power switch for ALTRO Power Regulator
CSR3[15:0]	0x14	RW	Yes	Status and Configuration *Default value = 0x2220* [15] This bit is set to 1 when the BC has completed the transaction with the mADC. It is reset at the beginning of every transaction. [14:8] Scevl timeout value. Max num of clks for each transaction on the ALTRO bus when BC is master. [7:0] rdclk / sclk warning ratio
DEBUG[1:0]	0x15	R	No	[1] Value of Slow Control data line [0] Value of test_mg input
CNTLAT	0x16	T	Yes	Latch L0, L2, SCLK counters
CNTCLR	0x17	T	Yes	Clear L0, L2, SCLK counters
CSR1CLR	0x18	T	Yes	Clear Error Status Register
ALRST	0x19	T	Yes	Reset all the ALTROs
BCRST	0x1A	T	Yes	Reset Board Controller to default values
STCNV	0x1B	T	Yes	Start Conversion / Readout monitor ADC
SCEVL	0x1C	T	Yes	Scan Event-length registers in all ALTRO channels
EVLRDO	0x1D	T	Yes	Start readout of Event Length Hitmap register.
UPDATEHV	0x1E	T	Yes	Update HV
BCVERSION[15:0]	0x20	R	N/A	Board Controller Version

Register name	Addr.	Type [2]	Allow Brcast	Description
VTS_HIGH[14:0]	0x21	R(W)	Yes*	Voltage Temperature Status register: [0] TEMP1 over th [1] D4V0 over th [2] D4V0C over th [3] D3V3 over th [4] D3V3C over th [5] TEMP2 over th [6] A6nV0 over th [7] A6nV0C over th [8] A6pV0 over th [9] A6pV0C over th [10] TEMP3 over th [11] A3V3 over th [12] A3V3C over th [13] A13V0 over th [14] A13V0C over th
VTS_LOW[14:0]	0x22	R(W)	Yes*	Voltage Temperature Status register: [0] TEMP1 under th [1] D4V0 under th [2] D4V0C under th [3] D3V3 under th [4] D3V3C under th [5] TEMP2 under th [6] A6nV0 under th [7] A6nV0C under th [8] A6pV0 under th [9] A6pV0C under th [10] TEMP3 under th [11] A3V3 under th [12] A3V3C under th [13] A13V0 under th [14] A13V0C under th
TH_HMGERR_HIGH[14:0]	0x23	R(W)	Yes*	Double hamming errors found in ADC high threshold memory .
TH_HMGERR_LOW[14:0]	0x24	R(W)	Yes*	Double hamming errors found in ADC low threshold memory.
HV_FB1[15:0]	0x25	R(W)	Yes*	Compared outputs from DAC for CSP [7:0] CSP 16 – CSP 23 [15:8] CSP 7 – CSP 0 0: DAC not set/Wrong 1: DAC set/Correct
HV_FB2[15:0]	0x26	R(W)	Yes*	Compared outputs from DAC for CSP [7:0] CSP 15 – CSP 8 [15:8] CSP 24 – CSP 31 0: DAC not set/Wrong 1: DAC set/Correct
HV_HVHMGERR1[15:0]	0x27	R(W)	Yes*	Double hamming errors for the following: [7:0] CSP 16 – CSP 23 [15:8] CSP 7 – CSP 0
HV_HVHMGERR2[15:0]	0x28	R(W)	Yes*	Double hamming errors for the following: [7:0] CSP 15 – CSP 8 [15:8] CSP 24 – CSP 31
ADC_DIFF[14:0]	0x29	RW	Yes	Sets which ADC values that should be treated as currents in the ADC value and ADC threshold memory. In practice this is a diff between V_{prev} and $V_{current}$ *Default Value: 0x5294*

Register name	Addr.	Type [2]	Allow Brcast	Description
ADC_DIFF_DIR [14:0]	0x2A	R(W)	Yes*	Sets the expected direction of the current for the current measurements given by ADC_DIFF register. 0: Current = $V_{previou}$ - $V_{current}$ 1: Current = $V_{current}$ – $V_{previou}$ *Default Value: 0x0080*
TRANS_CNT[15:0]	0x2B	R	No	[15:8] Number of acks sent to RCU [7:0] Number of strobes received from RCU
LAST_ADDR_MSB [7:0]	0x2C	R	No	[7:4] MSB of last valid address received. [3:0] MSB of last not valid address received
LAST_VALID_ADDR [15:0]	0x2D	R	No	LSB of last valid address received at the FEC
LAST_NOTVALID_ ADDR[15:0]	0x2E	R	No	LSB of the last not valid address received at the FEC.
ADC Min Threshold Memory[14:0]	0x30 – 0x3E	RW	Yes	Min Threshold for the ADCs [15:11] Hamming code [10] 0: Threshold for Voltage 1: Threshold for Current [9:0] Data value
ADC Max Threshold Memory[14:0]	0x40 – 0x4E	RW	Yes	Max Threshold for the ADCs [15:11] Hamming code [10] 0: Threshold for Voltage 1: Threshold for Current [9:0] Data value
ADC Data Memory[9:0]	0x50 – 0x5E	RW	Yes	Data values from the ADCs [9:0] Data value
HV DAC settings memory [14:0]	0x60 – 0x7F	RW	Yes	High voltage bias value for CSPs 0x60-0x67: CSP 23 down to CSP 16 0x68-0x6F: CSP 0 to CSP 7 0x70-0x77: CSP 8 to CSP 15 0x78-0x7F: CSP 31 down to CSP 24 [15:11] Hamming code [10] Don't care (not used) [9:0] Value to Write

Table 4-1: List of registers that can be accessed externally. Note: The registers marked with "R(W)" and Broadcast "Yes*" can be written to when unlock bit is set.

Memory location name	Addr.	Description
TEMP1-MIN_TH	0x30	Minimum Temperature Threshold for ADC IC13 *Default Data Value: 0x0 (disabled)*
D4V0_MIN_TH[	0x31	Minimum 4.0V Digital Voltage Threshold *Default Data Value: 0x1D8 (= 3.8 V)*
D4V0C_MIN_TH	0x32	Minimum 4.0V Digital Current Threshold *Default Value: 0x0 (disabled)* Alternatively Voltage level used for current calc.
D3V3_MIN_TH	0x33	Minimum 3.3V Digital Voltage Threshold *Default Data Value: 0x1C2 (= 2.9 V)*
D3V3C_MIN_TH	0x34	Minimum 3.3V Digital Current Threshold *Default Value: 0x0 (disabled)* Alternatively Voltage level used for current calc.
TEMP2-MIN_TH	0x35	Minimum Temperature Threshold for ADC IC15 *Default Value: 0x0 (disabled)*
A6nV0_MIN_TH	0x36	Minimum -6.0V Analog Voltage Threshold *Default Data Value: 0x170 (= -6.4 V)*
A6nV0C_MIN_TH	0x37	Minimum -6.0V Analog Current Threshold *Default Value: 0x0 (disabled)* Alternatively Voltage level used for current calc.
A6pV0_MIN_TH	0x38	Minimum 6.0V Analog Voltage Threshold *Default Data Value: 0x1E8 (= 5.6 V)*
A6pV0C_MIN_TH	0x39	Minimum 6.0V Analog Current Threshold *Default Value: 0x0 (disabled)* Alternatively Voltage level used for current calc.
TEMP3-MIN_TH	0x3A	Minimum Temperature Threshold for ADC IC14 *Default Value: 0x0 (disabled)*
A3V3_MIN_TH	0x3B	Minimum. 3.3V Analog Voltage Threshold *Default Value: 0x1C2 (= 2.9 V)*
A3V3C_MIN_TH	0x3C	Minimum 3.3V Analog Current Threshold *Default Value: 0x0 (disabled)* Alternatively Voltage level used for current calc
A13V0_MIN_TH	0x3D	Minimum 13.0V Analog Voltage Threshold *Default Value: 0x1D6 (= 12.6 V)*
A13V0C_MIN_TH	0x3E	Minimum 13.0V Analog Current Threshold *Default Value: 0x0 (disabled)* Alternatively Voltage level used for current calc

Table 4-2: ADC Minimum Threshold Value Memory. The conversion factors are given in Table 4-4

Memory location name	Addr.	Description
TEMP1-MAX_TH	0x40	Maximum Temperature Threshold for ADC IC13 *Default Data Value: 0xA0 (= 40°C)*
D4V0_MAX_TH[	0x41	Maximum 4.0V Digital Voltage Threshold *Default Value: 0x0 (disabled)*
D4V0C_MAX_TH	0x42	Maximum 4.0V Digital Current Threshold *Default Value: 0x00C (=0.36 A)* Alternatively Voltage level used for current calc.
D3V3_MAX_TH	0x43	Maximum 3.3V Digital Voltage Threshold *Default Value: 0x0 (disabled)*
D3V3C_MAX_TH	0x44	Maximum 3.3V Digital Current Threshold *Default Value: 0x011 (= 0.73 A)* Alternatively Voltage level used for current calc.
TEMP2-MAX_TH	0x45	Maximum Temperature Threshold for ADC IC15 *Default Data Value: 0xA0 (= 40°C)*
A6nV0_MAX_TH	0x46	Maximum -6.0V Analog Voltage Threshold *Default Value: 0x0 (disabled)*
A6nV0C_MAX_TH	0x47	Maximum -6.0V Analog Current Threshold *Default Value: 0x00F (=0.44 A)* Alternatively Voltage level used for current calc.
A6pV0_MAX_TH	0x48	Maximum 6.0V Analog Voltage Threshold *Default Value: 0x0 (disabled)*
A6pV0C_MAX_TH	0x49	Maximum 6.0V Analog Current Threshold *Default Value: 0x016 (= 0.764 A)* Alternatively Voltage level used for current calc.
TEMP3-MAX_TH	0x4A	Maximum Temperature Threshold for ADC IC14 *Default Data Value: 0xA0 (= 40°C)*
A3V3_MAX_TH	0x4B	Maximum. 3.3V Analog Voltage Threshold *Default Value: 0x0 (disabled)*
A3V3C_MAX_TH	0x4C	Maximum 3.3V Analog Current Threshold *Default value: 0x014 (= 0.858 A)* Alternatively Voltage level used for current calc
A13V0_MAX_TH	0x4D	Maximum 13.0V Analog Voltage Threshold *Default Value: 0x0 (disabled)*
A13V0C_MAX_TH	0x4E	Maximum 13.0V Analog Current Threshold *Default Value: 0x00F (= 0.334 A)* Alternatively Voltage level used for current calc

Table 4-3: ADC Maximum Threshold Value Memory

Memory location name	Addr.	Description	Conv. factor
TEMP1	0x50	Temperature for ADC IC13	0.25°C * ADC counts
D4V0	0x51	4.0V Digital Voltage	8.04mV * ADC counts
D4V0C	0x52	4.0V Digital Current Alternatively Voltage level used for current calc.	29.8mA * ADC counts
D3V3	0x53	3.3V Digital Voltage	6.44 mV * ADC counts
D3V3C	0x54	3.3V Digital Current Alternatively Voltage level used for current calc.	42.9 mA * ADC counts
TEMP2	0x55	Temperature for ADC IC15	0.25°C * ADC counts
A6nV0	0x56	-6.0V Analog Voltage	4.88mV * ADC counts / 1000 - 8.2V
A6nV0C	0x57	-6.0V Analog Current Alternatively Voltage level used for current calc.	29.3mA * ADC counts
A6pV0	0x58	6.0V Analog Voltage *Default Data Value: 0x1E8 (= 5.6 V)*	11.4 mV * ADC counts
A6pV0C	0x59	6.0V Analog Current Alternatively Voltage level used for current calc.	34.73 mA * ADC counts
TEMP3	0x5A	Temperature for ADC IC14	0.25°C * ADC counts
A3V3	0x5B	3.3V Analog Voltage	6.44 mV * ADC counts
A3V3C	0x5C	3.3V Analog Current Alternatively Voltage level used for current calc	42.9 mA * ADC counts
A13V0	0x5D	13.0V Analog Voltage	26.8 mV * ADC counts
A13V0C	0x5E	13.0V Analog Current Alternatively Voltage level used for current calc	22.3 mA * ADC counts

Table 4-4: ADC Value Memory. Please note that the current conversion factors are only correct if current mode is set in the Threshold register for given ADC value

5 Functional Requirements

5.1 Functional Overview

The main features of the PHOS Board Controller are:

- Control the ALTRO bus and the GTL drivers on the FEC.
- Read temperatures, voltages and currents on the board and verify them against locally stored thresholds. Flag if an error situation has occurred on the board.
- Set the output of the DACs that control the bias of the high voltage section.
- Read Eventlength registers in ALTRO and push hitmap data to the RCU when Sparse Readout Mode is selected.

In addition the Board Controller is able to set the FEC in standby mode by turning off the power supplies and disabling the clock and bus transactions.

Hamming is per default enabled, while continuously checking ADC values are not. Before turning on continuously check on ADC values all threshold should be set correctly. This includes adc_diff and adc_diff_dir registers – which in practice decides whether one should read current or voltage from the ADC. These memory location are set correctly by default. The Hamming encoding spans also over bit 10:0 in the threshold and dac registers.

The hamming decoding and, if needed, correction is done every time the values from the memories are read by internal logic. When reading the ADC values they are checked against the thresholds given in the min max threshold memories if the threshold memory location value is unequal to 0. If 0, this means that the test is disabled. The VTS registers (and CSR1) are set after the configurable number (1-3) of times a threshold is violated for each value. The number of times is set by CSR2 bit 8:7, and is default set to 1. If a given threshold is unequal to 0, this means that a violation of this threshold will trigger the interrupt line to the RCU.

5.2 Main Functional Changes From PCM v2.0 (HUST)

- Removal of USB communication
- Removal of Board ID register
- Hamming encoding and TMR of static registers.
- Some register remapping.
- Thresholds and ADC values stored in memories
- Removed Testmode

5.3 Project Setup

The complete design is checked into the CVS Repository of the Experimental Nuclear Physics group, University of Bergen[3], under the folder /vhdlcvs/phos_bc/.

File	Folder	Description
bc.cr.mti	/	Questasim project file.
bc.mpf	/	Questasim configuration file
func.do	/	Executes functional simulation
ppr.do	/	Executes post place and route simulation.
fmd.rar	/code-ref/	Fmd project

[3] http://web.ift.uib.no/kjekscgi-bin/viewcvs.cgi/

File	Folder	Description
PCM2.0_061130_2.rar	/code-ref/	HUST PCM2.0 project
PCM2.0_061130_code.rar	/code-ref/	HUST PCM2.0 verilog
PHOS BC specification.doc	/docs/	PHOS BC v3.0 spec
PHOS_BC_specification_v3.1.doc	/docs/	PHOS BC v3.1 spec (this document)
conv factor.xls	/docs/	Hamming encoding basis
Graphics.vsd	/docs/graphics	Visio file with graphics
*	/docs/ref/	Documents that are used as reference for the design
*	*/Quartus_project/*	The files generated/used by Quartus. Important files are: • bc.pof, bc.sof: configuration files • bc.vho: Simulation model • bc.pin: Pinning file
bc.vhd	/vhdl/	Top level
bc_tb2.vhd	/vhdl/	Testbench
register_config.vhd	/vhdl/	Package
altro_sw_mask_in.vhd	/vhdl/ altro_sw_mask_in	See chapter 5.4.3
alprotocol_if.vhd	vhdl\ altrobusinterface	See chapter 5.4.4 (obsolete v3.3)
altrobusinterface.vhd	vhdl\ altrobusinterface	See chapter 5.4.4
altrobusinterface_tb.vhd	vhdl\ altrobusinterface	See chapter 5.4.4 (not updated)
interfacebus.vhd	vhdl\ altrobusinterface	See chapter 5.4.4 (obsolete v3.3)
drivers.vhd	/vhdl/drivers	See chapter 5.4.2
signals_driver.vhd	/vhdl/drivers	See chapter 5.4.2
transceivers_driver.vhd	/vhdl/drivers	See chapter 5.4.2
ch_counter.vhd	/vhdl/evl_man/	See chapter 5.4.5
evl_man.vhd	/vhdl/evl_man/	See chapter 5.4.5
evlreg_trsf.vhd	/vhdl/evl_man/	See chapter 5.4.5
read_evl.vhd	/vhdl/evl_man/	See chapter 5.4.5
hvdac.vhd	/vhdl/hvdac	See chapter 5.4.10
hvdac_tb.vhd	/vhdl/hvdac	See chapter 5.4.10
adc_rom.cmp	/vhdl/interface_adc	See chapter 5.4.9
adc_rom.vhd	/vhdl/interface_adc	See chapter 5.4.9
interface_adc.vhd	/vhdl/interface_adc	See chapter 5.4.9
interface_adc_tb.vhd	/vhdl/interface_adc/	See chapter 5.4.9 (not updated)
master.vhd	/vhdl/interface_adc	See chapter 5.4.9
master_sm.vhd	/vhdl/interface_adc	See chapter 5.4.9
ROM.hex	/vhdl/interface_adc	See chapter 5.4.9
ROM.mif	/vhdl/interface_adc	See chapter 5.4.9
rom.vhd	*/vhdl/interface_adc*	*Obsolete*
sequencer.vhd	/vhdl/interface_adc	See chapter 5.4.9
serializer.vhd	/vhdl/interface_adc	See chapter 5.4.9
interfacedec.vhd	/vhdl/interfacedec	See chapter 5.4.7
adc_ram.cmp	/vhdl/registers	See chapter 5.4.8
adc_ram.vhd	/vhdl/registers	See chapter 5.4.8
counters.vhd	*/vhdl/registers*	*obsolete*
dac_ram.cmp	/vhdl/registers	See chapter 5.4.8
dac_ram.vhd	/vhdl/registers	See chapter 5.4.8
df_adcval.hex	/vhdl/registers	See chapter 5.4.8
df_adcval.mif	/vhdl/registers	See chapter 5.4.8
df_dac.hex	/vhdl/registers	See chapter 5.4.8
df_dac.mif	/vhdl/registers	See chapter 5.4.8
df_thhigh.hex	/vhdl/registers	See chapter 5.4.8

File	Folder	Description
df_thhigh.mif	/vhdl/registers	See chapter 5.4.8
df_thlow.hex	/vhdl/registers	See chapter 5.4.8
df_thlow.mif	/vhdl/registers	See chapter 5.4.8
dstb_counter.vhd	*/vhdl/registers*	*obsolete*
hamming_decoder.vhd	/vhdl/registers	See chapter 5.4.8
registers.vhd	/vhdl/registers	See chapter 5.4.8
registers_block.vhd	/vhdl/registers	See chapter 5.4.8
sclk_counter.vhd	/vhdl/registers	See chapter 5.4.8
th_ram.hex	*/vhdl/registers*	*obsolete*
th_ram.mif	*/vhdl/registers*	*obsolete*
triggercounter.vhd	/vhdl/registers	See chapter 5.4.8
vtc_status.vhd	/vhdl/registers	See chapter 5.4.8
fec_address.vhd	/vhdl/slave	See chapter 5.4.6
sel_signals.vhd	/vhdl/slave	See chapter 5.4.6
serializer_bc.vhd	/vhdl/slave	See chapter 5.4.6
slave.vhd	/vhdl/slave	See chapter 5.4.6
slave_rx.vhd	/vhdl/slave	See chapter 5.4.6
slave_tb.vhd	/vhdl/slave	See chapter 5.4.6 (not updated)
slave_tx.vhd	/vhdl/slave	See chapter 5.4.6
ad7417.vhd	/vhdl/testbench	See chapter 6.4
ad7417_tb.vhd	/vhdl/testbench	See chapter 6.4 (not updated)
max5308dac.vhd	/vhdl/testbench	See chapter 6.4
rcu_synthesis.vhd	/vhdl/testbench	See chapter 6.4
tb_pkg.vhd	/vhdl/testbench	See chapter 6.4
tb_pkg_phosBC.vhd	/vhdl/testbench	See chapter 6.4
tb_txt_util.vhd	/vhdl/testbench	See chapter 6.4

Table 5-1: Files checked in in the CVS repository.

5.3.1 Software

Editor: ConTEXT v0.98.5
Simulation: Questasim 6.1d
Synthesis and Place and Route for test: Quartus II Version 6.0 Build 178

5.4 Functional Details

5.4.1 Main Output Signals

Signal Name	Explanation
rdoclk_en	Enables the distribution of the system clock to all the ALTROs.
adcclk_en	Enables distribution of the sampling clock to the ADCs.
Ackn	ALTRO bus acknowledge line
Error	ALTRO bus error line
trsf	ALTRO bus transfer line
dstb	ALTRO bus data strobe
bcout_add[4:0]	Masked address to ALTRO
altrops_en	ALTRO power supply enable
biasps_en	BIAS power supply enable
shaperps_en	SHAPER power supply enable
oeab_l	ALTRO bus GTL driver enable
oeab_h	ALTRO bus GTL driver enable
oeba_l	ALTRO bus GTL driver enable
oeba_h	ALTRO bus GTL driver enable
ctr_in	CONTROL bus GTL driver enable
ctr_out	CONTROL bus GTL driver enable
rcu_sda_out	Slow control data out
bc_int	Interrupt line
sensor_scl	ADC I2C interface
sensor_sda	ADC I2C interface
convst	ADC interface convert start
dac_clk_en	DAC serial interface
dac_sclk	DAC serial interface
dac_ldac	DAC serial interface
dac_sel[3:0]	DAC serial interface
dac_din[3:0]	DAC serial interface

Table 5-2: Main output signals with explanations

5.4.2 Drivers (Glue Logic)

The purpose of the Drivers Module is twofold. Firstly it is in charge of driving the direction of the GTL bus drivers for the ALTRO bus and the CONTROL bus, and secondly it tristates the signals on the bus since the Front End Bus is common for all FECs connected to the RCU.

5.4.3 ALTRO Switch Mask In

The ALTRO Switch Mask In Module is a combinatorial masking of ALTRO bus input signals and internal mask bits. The internal mask bits are the power supplies enable bits from CSR3. If the ALTRO power supplies are turned off, all communication with the ALTROs are masked out, as well as error/status information concerning the ALTRO to the RCU. If the BIAS and the SHAPER power supplies are turned off, error/status information concerning the ALTRO to the RCU are masked out.

Additionally it adds metastability filter on the strobe and acknowledge signal.

5.4.4 ALTRO Interface

As given by the name the ALTRO interface decoded the information coming on the ALTRO bus. It listens for a positive edge on the strobe and then looks at the 20

most significant bits in the ALTRO bus where it decodes the address. Decoding the address essentially means:

- Decoding parity bit to detect any parity error → Not valid transaction
- Check the bcast bit. If asserted, no ack should be given
- Check the ALTRO_BC bit that selects either ALTRO or BC.
- Check the hardware address.

If all these are correct for the given FEC and Board Controller, the 7 least significant bits are handed over to the registers module for decoding and the transaction is ack. The transaction will be acked no matter if the given register addresses does exist or not.

Two chip select signals are being decoded, one for if it is a valid board controller transaction and one if it is a valid ALTRO transaction. These CS are used by the Drivers Module to control the GTL bus drivers.

Since the ALTRO bus has a problem with occasional timeouts, debug information was from version 3.4 made available in 4 registers. This information gives a notice on how many strobes the BC has seen from the FEC and how many has actually been acked. In addition the last 20 bit address being acked on the ALTRO bus is stored, as well as the last 20 bit address that has not being acked is stored. This can be used for debugging why the given BC does not ack on registers where it should've done so.

Version 3.4 also introduced more robustness measures concerning the bus such as meta-stability filters on all control signals and address information and an extra wait state to be sure that the data/address are stable when the are decoded.

5.4.5 Event-length Manager

The Event-length Manager is used in Sparse Readout Mode, and the purpose of the Event-length Manager is to scan the Event Length register of each ALTRO channel to see if it contains data (zero-suppression must be enabled). The module builds a hitmap register that has one bit for each channel that has event length unlike zero. The hitmap register is pushed to the RCU in the same manner as the ALTRO pushes data.

The Event-length manager listens on two commands from the register block (decoded from the RCU)

- Scan Eventlength (SCEVL)
- Event-length Readout (EVL_RDO)

The first command starts to read the event length register of all the ALTRO channels. To do this it needs to take control over the ALTRO bus and become the bus master. It then isolates the board from the RCU for a short while wile doing this. When the hitmap register is filled, it waits for the event-length readout command on which it pushes the data to the RCU.

Please note: The pushing of the data uses a gated version of the rdoclk as dstb as this is done originally. This is not an ideal solution and might lead to glitches and misbehaviour of Sparse Readout Mode. It would be better to let the dstb run on half the speed making a registered output as the dstb.

5.4.6 Slow Control Slave Interface

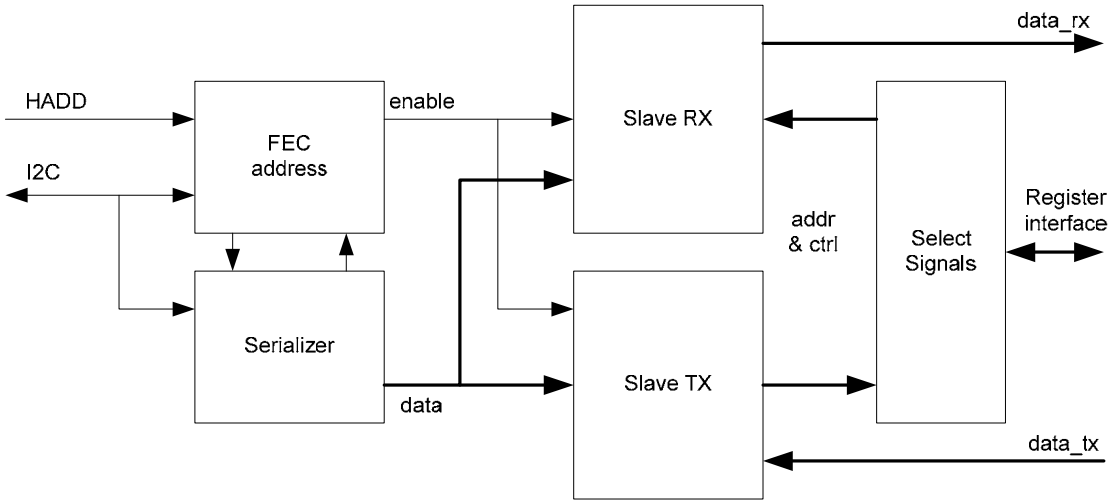


Figure 5-1: Sketch of Slow Control Interface.

The Slow Control Interface consists of five sub-modules:

- **FEC address**: A state machine listening for the I2C start condition, and then decode the first byte transferred. If this byte contains the address of this card (bits 5 to 1), or is broadcast (bit 6), the state machine acknowledges the request. If the request is a write request (bit 0 is low), then the receiver is started. If the request is a read request (bit 0 high), the transmitter is started.
- **Serializer:** A shift-register of 1 byte. Serial input is the I2C bus serial data. Parallel input is register values from the BC. The parallel output is used by all state machines to get the needed data. The serial output is used by the transmitter to serialize the parallel input to the master of the I2C bus.
- **Slave RX:** Receives one 1 byte, and compares the lower 7 bit to valid instructions. If the instruction is correct, then the state machine acknowledges the request and reads 2 more bytes. The instruction code and 16bit data is output for handling by the BC.
- **Slave TX:** Receives one 1 byte, and compares the lower 7 bit to valid instructions. If the instruction is correct, then the state machine acknowledges the request. The instruction code is sent off to the BC, and valid data should be returned. The transmitter then transfers the data in 2 1-byte blocks to the master of the I2C bus.
- **Select Signals:** A multiplexer. Depending on the state of the various state machines, this mux chooses the control inputs to the serializer, and the address returned to the BC.

In version 3.2, this module has been upgraded. Now this module will decode all that is received on the slow control bus no matter if the card address is correct or not. This is done to ensure that a fake start condition can not be detected when a different board is addressed and busy with a transaction. If the card address is not correct, a mask bit has been set to mask both the sda_out data line and the internal write enable. In addition the RX and TX module have now timeout counters for each byte received. It takes ~78 clks (system clock) to received, and for convenience the modules times out and goes back to idle when 128 clks has passed.

5.4.7 Interface Decoder

The Interface Decoder is a state machine that decodes the information on the ALTRO bus or on the I2C bus. It sets error status information for the received addresses and decodes the command (trigger) addresses into single command lines.

5.4.8 Registers

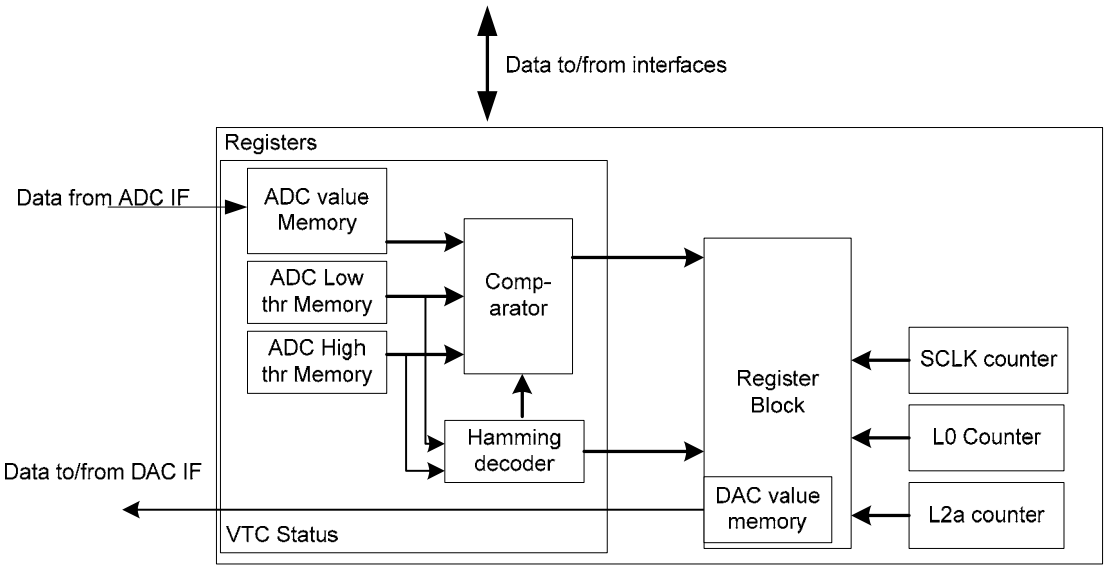


Figure 5-2: Registers module with all the different submodules and memories given

The Registers Module consists of 3 counters, a Register Block and a VTC (Voltage, Temperature and Current) Status Module as shown in Figure 5-2. The slow clock counter counts the sampling clock for housekeeping purposes. In addition there are two trigger counters that count the L0 and L2a triggers received.

The Register Block is the important module and holds all the registers in the Board Controller. All error states are tested for in this module and the correct error bits are set. The interface to the module is a fully synchronous interface with data, address and a write enable signal. The DAC value memory are hamming coded and the validity of the contents of these memories will be checked whenever the memories are read by internal logic (not the interfaces).

The CSR0, CSR2, CSR3, ADC_DIFF and ADC_DIFF_INT registers are protected against single event upsets by the use of TMR (Triple Modular Redundancy) and voting logic.

The DAC value memory holds all the values to be written to the DAC interface. The DAC interface itself has a hamming decoder performing a hamming check before the value is written to the DAC. If a single bit error is found the DAC value memory is updated with the correct value. If a double error is found then this is notified in the DAC hamming error register and the given DAC is not written to

The VTC Status Module holds all memories related to the read out of ADC values. Everytime the adc_we signal is asserted from the ADC interface, the comparator verifies if the value delivered by the ADC is within the thresholds. The thresholds are also hamming tested every time they are read by internal logic if hamming is enabled for the BC. There is a separate state machine (comparator) that tests whether the read back ADC value is within the threshold given in the threshold memory. If the hamming decoder is enabled then the threshold value is validated first. In case of a single bit error, the ADC threshold register will be updated with the correct value. In case of a double bit error this is notified in the threshold hamming

error register. This functionality is executed whenever the ADC interface writes a value to the ADC value memory. The ADC threshold memory and the ADC value memory have the same addressing, meaning that there is a one to one correspondence between the values in a given address.

If the threshold test fails (the read back value exceeds the value given in threshold) an error counter is counted up for the given position. This counter can trigger the interrupt to go off at a configurable number of times (from 1 to 3) the same error situation has occurred in a row. Default value of this limit is set to 1, to match previous versions of the BC. This is implemented so that it is possible to avoid that a single event upset or any other error in the reading back of ADC value will fire an interrupt that might turn off the complete board when it is false alarm.

The ADC_DIFF and ADC_DIFF_INT sets the mode of which the adc value should be perceived. If ADC_DIFF equals 0, the value read is tested directly against the given threshold value and stored into the ADC value memory. If ADC_DIFF equals 1, it means that the difference between the currently returned value and the previous read value is tested against the threshold and stored in the ADC value memory. Since the difference between the previous and the current value is effectively a measurement for the current when used as in PHOS, and additional register ADC_DIFF_INT is used to set the correct expected current direction. The VTC error counter is counted up for the given position if the current flows the opposite way of what it is expected to do. .

5.4.9 ADC Interface

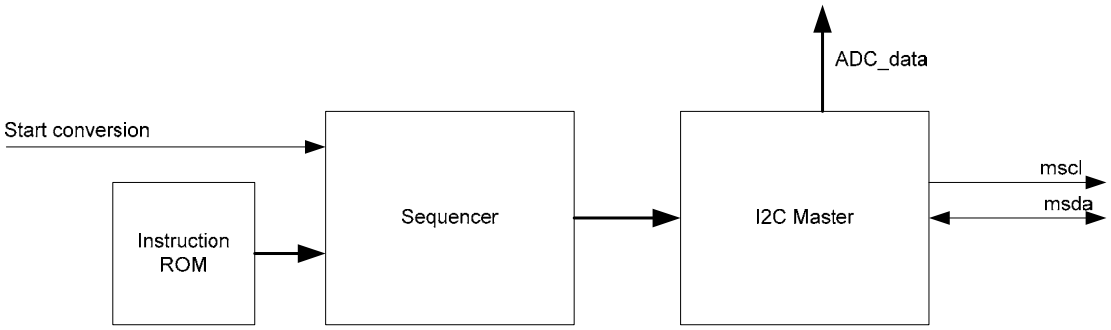


Figure 5-3: Sketch of the ADC interface that is used for monitoring of temperatures, currents and voltages on the Front End Card.

The main purpose of the ADC interface is to read the currents, voltages and temperatures on the different parts of the board. There are three ADCs of type AD7417 [2] placed in different areas on the board, and these are controlled using a standard I2C bus protocol. All the instructions for reading/writing are placed in a ROM. If “start conversion” is high when the Sequencer is in the idle state, a complete readback of all values of the ADCs are initiated. The sequencer is level sensitive of “Start Conversion”, meaning that if one wants to continuously readback all the data, this is done by setting this input constantly high. The ADCs offer a possibility to let them verify the temperature against a programmable threshold set a over-temperature flag (OTI). This is not used by the Board Controller since this functionality is kept internal in the BC firmware.

For the conversion factors of the different ADC values please see the adc value adress table (Table 4-4)

ADC	Address	Location on FEE
IC13	“000”	Top: Between ALTRO 0 and ALTRO 2
IC15	“001”	Top: Power Regulator Area
IC14	“010”	Bottom: Between ALTRO 3 and ALTRO 4

Table 5-3: ADCs used for monitoring in PHOS FEE.

5.4.10 DAC Interface

The DAC interface is responsible for setting the BIAS voltages on all the CSPs on the FEC: There all together 4 DACs on the board, connected with 4 separate serial buses. Each DAC has 8 channels. Adding it up it will be all together 32 DAC settings to be made.

The DAC itself has a serial interface, where 16 bits needs to be shifted to the DAC to set one channel. The 4 DACs have 4 separate data out, data in and chip select signals and one common clock line. On the data out line from the DAC the bits that was shifted in last time is directly shifted out. In the DAC interface this is used to verify that the bits shifted in the last time is correctly received by the DAC.

The DAC interface consists of a state machine that gets out of idle state when the HV update signal is high or when the continuously updating is selected. It then read one by one value in the DAC Memory in the Registers Block. If hamming decoding is enabled it immediately verifies the contents of the addressed DAC Memory location. If a single bit error is found it corrects the given DAC Memory location and goes on to shift the value to the addressed DAC. If it is a double error this is reported in a status register and the given DAC channel is skipped.

When shifting in data it does it in the following sequence:

- data to all 8 channels in one DAC
- NOOP command
- Update output command
- Power up Command

The NOOP command is needed to verify that the last channel is correctly shifted in. Only the channels that are correctly shifted in and are without double bit hamming errors are updated. The Power up command is sent to all channels, pulling the output of the DAC to 0V for the channels that are in error.

CSP	DAC addr	DAC channel/code	External address
CSP23	"00"	"0010"	0x60
CSP22	"00"	"0011"	0x61
CSP21	"00"	"0100"	0x62
CSP20	"00"	"0101"	0x63
CSP19	"00"	"0110"	0x64
CSP18	"00"	"0111"	0x65
CSP17	"00"	"1000"	0x66
CSP16	"00"	"1001"	0x67
CSP0	"01"	"0010"	0x68
CSP1	"01"	"0011"	0x69
CSP2	"01"	"0100"	0x6A
CSP3	"01"	"0101"	0x6B
CSP4	"01"	"0110"	0x6C
CSP5	"01"	"0111"	0x6D
CSP6	"01"	"1000"	0x6E
CSP7	"01"	"1001"	0x6F
CSP8	"10"	"0010"	0x70
CSP9	"10"	"0011"	0x71
CSP10	"10"	"0100"	0x72
CSP11	"10"	"0101"	0x73
CSP12	"10"	"0110"	0x74
CSP13	"10"	"0111"	0x75
CSP14	"10"	"1000"	0x76
CSP15	"10"	"1001"	0x77
CSP31	"11"	"0010"	0x78
CSP30	"11"	"0011"	0x79
CSP29	"11"	"0100"	0x7A
CSP28	"11"	"0101"	0x7B
CSP27	"11"	"0110"	0x7C
CSP26	"11"	"0111"	0x7D
CSP25	"11"	"1000"	0x7E
CSP24	"11"	"1001"	0x7F

Table 5-4: HVDAC setup, including mapping of CSP.

5.4.11 Hamming Code / Hamming Decoder Module

The hamming decoder module is used in both the Register Block for ADC thresholds and in the DAC interface for DAC values. If a single error is found it is reported and corrected. If a double error is found it is just reported.

The hamming code is generated the following way:

$h(0) = d(0) \oplus d(1) \oplus d(3) \oplus d(4) \oplus d(6) \oplus d(8) \oplus d(10)$

$h(1) = d(0) \oplus d(2) \oplus d(3) \oplus d(5) \oplus d(6) \oplus d(9) \oplus d(10)$

$h(2) = d(1) \oplus d(2) \oplus d(3) \oplus d(7) \oplus d(8) \oplus d(9) \oplus d(10)$

$h(3) = d(4) \oplus d(5) \oplus d(6) \oplus d(7) \oplus d(8) \oplus d(9) \oplus d(10)$

$h(4) = h(0) \oplus h(1) \oplus h(2) \oplus h(3) \oplus d(0) \oplus d(1) \oplus d(2) \oplus d(3) \oplus d(4) \oplus d(5) \oplus d(6) \oplus d(7) \oplus d(8) \oplus d(9) \oplus d(10)$

Where h is the 5 bit hamming vector and d is the 11 bit data vector. Please note that when hamming decoding is enabled, the threshold registers and the DAC value registers must be filled in with hamming code and data for the firmware to act as expected. If hamming decoding is disabled only the last 10 bits in these register

matters – the other bits are don't care. The formats of these registers are given in Table 4-1.

5.4.12 Optional Functionality

If it should be of any interest to store board specific data, for instance the serialnumber of the board, on the 24LC256 External Flash Memory, an I2C master can be made in the BC firmware that gives access to this device. The I2C bus is shared between the USB chip, the Flash Device and the Board Controller, of which the latter is us originally thought to be used as a slave. Since I2C master device on the USB chip has open drain outputs it should not be any electrical constraints preventing the BC from being the master, but investigations must be done to verify that the BC and the USB will not access the I2C bus at the same time for any reason. This has a very low priority

6 Other requirements

6.1 Clock strategy

40 MHz System Clock.
10 MHz Sampling Clock counted

6.2 Reset strategy

Asynch reset, negative polarity

6.3 Power strategy

N/A

6.4 Test strategy

6.4.1 Functional and Post Place and Route Verification

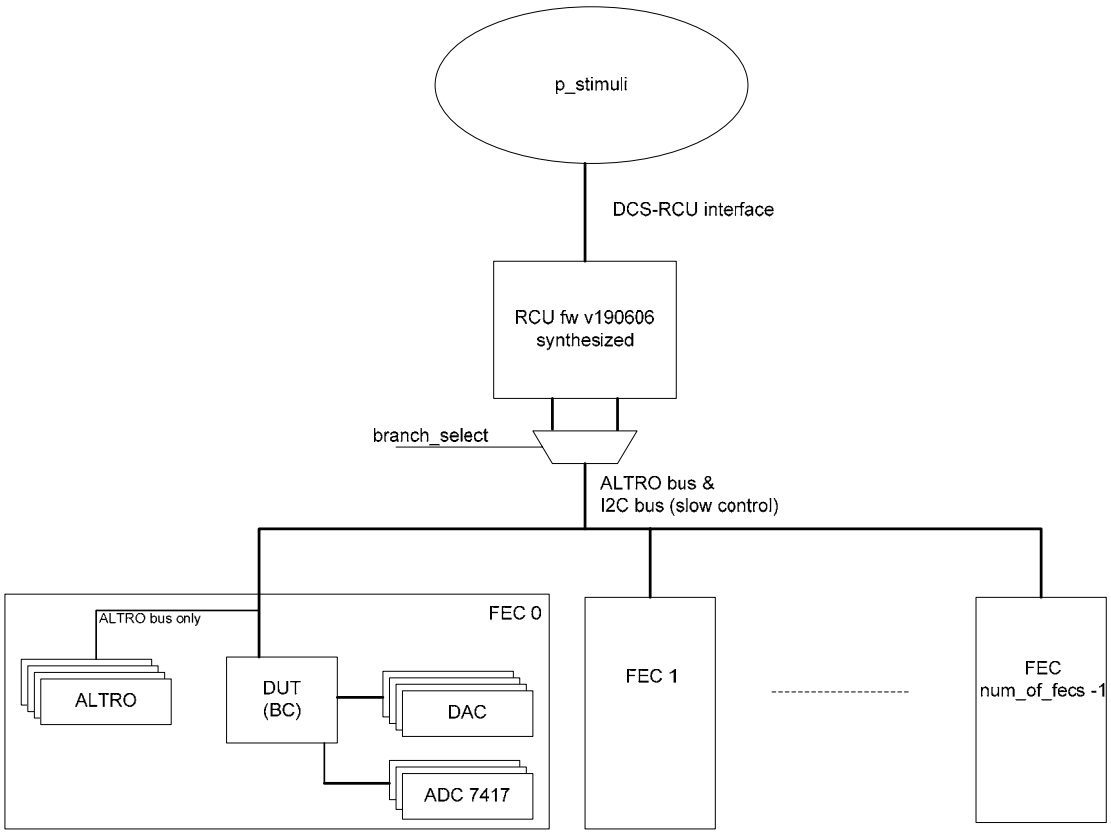


Figure 6-1: Testbench setup for functional verification and post place and route simulation.

The design has been verified both functionally and post place and route with Questasim using the testbench setup as given in Figure 6-1. The process p_stimuli uses procedures to read/write using the DCS interface on the RCU. The RCU Module is a synthesized simulation model generated by Xilinx ISE, based on RCU firmware version 190606. The Board Controller (DUT) also connects to simple simulation models of the ADCs (a modified opensource I2C slave simulation model) and the DACs that reports to the log when they are being accessed. The simulation model for the ALTRO is a slightly modified simulation downloaded from the TPC FEE

webpages[4]. Two generic variables makes it possible to choose between functional or post place and routes simulation, as well as the number of front end cards to include. The front end cards will be placed on the same branch. The testbench is semi-selftestable. Some functions – as the updating of the DACs and doing ADC readback, are done by simply inspecting the log afterwards. Other functionality is verified by inspection, such as the setting of the interrupts etc.

6.4.2 Functional Coverage

Functional Coverage has not been done.

6.4.3 Hardware Verification

The design has been functionally verified in hardware with a setup consisting of one RCU and one FEC connected to FEC address 9 on branch A. The RCU fw version 181206 and DCS board fw version 2.7. The ALTROs on the FEC is broken so it has not been possible to test if the BC affects data readout. But is has been verified that register access is no problem.

Cases that has been functionally verified is:

- Altro bus communication
- Slow Control communication
- DAC update on command and continuously
- ADC readback on command and continuously
- Single and double hamming error
- Interrupt handling from RCU
- Altro communication – see that BC does not mingle with the data bus.
- Trigger counters while doing data readout
- Broadcast

In addition a stress test has been performed of the Slow Control Interface and the ALTRO IF. On the given test-setup given here more that 1000000 transaction on the ALTRO bus went without error, and more than 250000 complete transmissions (4xbytes) on the Slow Control bus did the same.

On the complete module, it has been functionally verified that Slow Control communication works with more than one FEC attached and powered. Concerning the ALTRO bus protocol, certain timeout situation happens at a approximate rate of 1 per mill of all transactions. The reason for these timeout situations is that the Board Controller does not see a valid address when it occurs, and then does not ack. This has been tested with all FECs powered and with only one FEC powered and the result is the same. Since this problem has not been seen at all on the test setup in the lab, only at the module, the conclusion is that the Board Controller is performing as it should, but the error is somewhere externally.

[4] http://ep-ed-alice-tpc.web.cern.ch/ep-ed-alice-tpc/

7 Physical implementation

7.1 Technology

Device: Altera ACEX1K EP1K100QC208-3

7.2 Logic Synthesis

7.2.1 Static timing analysis

Type	Slack	Required Time	Actual Time	Failed Paths
Clock Setup: 'rdoclk'	0.800 ns	40.00 MHz (period = 25.000 ns)	41.32 MHz (period = 24.200 ns)	0
Clock Hold: 'rdoclk'	1.300 ns	40.00 MHz (period = 25.000 ns)	N/A	0
Recovery: 'rdoclk'	14.900 ns	26.200 ns	11.300 ns	0
Removal: 'rdoclk'	2.900 ns	2.800 ns	5.700 ns	0
Clock Setup: 'adcclk'	43.400 ns	20.00 MHz (period = 50.000 ns)	151.52 MHz (period = 6.600 ns)	0
Clock Hold: 'adcclk'	1.300 ns	20.00 MHz (period = 50.000 ns)	N/A	0
Recovery: 'adcclk'	44.300 ns	55.000 ns	5.700 ns	0
Removal: 'adcclk'	4.000 ns	7.600 ns	11.600 ns	0
Total number of failed paths				0

Table 7-1: Worst path timing information

7.2.2 Area estimates

Compilation Hierarchy	Logic Cells	LC Registers	Memory Bits	Pins
|bc	3421	1459	4096	146
|altro_sw_mask_in:mask|	20	11	0	0
|altrobusinterface:bus_interface|	141	122	0	0
|drivers:driv|	82	24	0	0
|evl_man:eventlength_manager|	197	137	0	0
|hvdac:DAC_interface|	610	196	0	0
|interface_adc:ADC_interface|	194	105	2816	0
|interfacedec:decoder|	56	0	0	0
|registers:regs|	1814	709	1280	0
|slave:slow_control_if|	297	151	0	0

Table 7-2: Area estimates giving total usage of resources and how much is used by the sub-modules.

8 Installation Log

This chapter gives a short overview on the installation of the Board Controller of 1st PHOS module.

8.1 Installation Status

The installation was done on the 29. – 30. of October 2007 on the complete module. The only board that was not upgraded is FEC B-12 on dcs0281, due to a problem with the JTAG chain. This board has still version 1.2 of PCM as made by HUST. It also is worth noticing that FEC A-12 on dcs0280 needs to have 3.3V externally fed to the JTAG chain because of a problem most probably related to a broken wire. This very cable has two connectors on it (as of 31. October 2007), and pin 2 is GND and pin4 is VCC.

All 4 RCUs has been verified in the sense that the version number has been read back from all FECs with success. One RCU has been more extensively tested to debug the problem with the ALTRO bus timeouts (see chapter 6.4.3).

8.2 Software

There are a couple of things that are interesting for the software developer and this chapter is an attempt to summarize them shortly.

1. The software must handle different mapping of various registers, since there is one board that still have the old version.
2. When doing writes and reads over the ALTRO interface always check that the transaction didn't timeout. This will be easier with the new RCU firmware version to handle in a fault proof manner.
3. Remember to verify certain status registers after doing various actions. An example can be the setting of the DAC values. Then there are 4 registers that should be checked – the High Voltage Feedback Registers giving which channels has been set, and the Hamming error registers in case hamming encoding is included.
4. If Hamming encoding is enabled, all values written the DAC memories and the ADC Threshold Memories must be written with correct hamming code. The calculation for the hamming code is included in this document.
5. Continuously readback of ADC values is not turned on by default and must be done if this is wanted.
6. Same accounts for continuously updating the DACs from the values stored in the memory.
7. Every once in a while, for instance this can be done every time the ADC values are read back, all kinds of status registers must be read back and verified is correct.